REPRESENTACIONES BÁSICAS, APLICACIÓN DETERMINÍSTICA Y PROBABILÍSTICA DE OPERADORES DE CROSSOVER EN ALGORITMOS GENÉTICOS CON ADAPTACION DINAMICA DE PROBABILIDADES.

VILANOVA G., MARQUEZ E., PANDOLFI D., MAC DONALD E., LORENZETTI D.[1]

Proyecto: Computación Evolutiva aplicada a problemas de Optimización en Ingeniería
División Tecnología
Universidad Nacional de la Patagonia Austral
Unidad Académica Caleta Olivia
Acceso Norte Ruta 3 CP (9011) Caleta Olivia - Santa Cruz - Argentina
E-mail: gvilanov@satlink.com emarquez@uaco.unpa.edu.ar
dpandolfi@uaco.unpa.edu.ar
ermd@uacounpa.edu.ar dlorenzetti@uaco.unpa.edu.ar
TE 54 0297 4550654 - FAX 54 0297 4854888

RESUMEN.

La adaptación de parámetros y operadores es una de las más importantes y promisorias áreas de investigación en la computación evolutiva. La idea es ajustar el algoritmo al problema mientras se resuelve el mismo.

En los algoritmos genéticos (AGs) no solo es necesario elegir la representación y los operadores para el problema, sino que también debemos elegir valores de parámetros y probabilidades de operadores del (AG) de manera tal que éste encuentre la solución y de manera eficiente. El proceso de encontrar *'a mano'* valores apropiados de parámetros y probabilidades de operadores para un AG que afecten la performance del algoritmo de una manera significativa, es una tarea que implica consumo de tiempo y esfuerzo considerable. Ello ha motivado la automatización de dicho proceso.

En los (AGs), las soluciones tentativas del espacio de búsqueda son codificadas o representadas tradicionalmente por medio de cadenas binarias, el *Gray coding* es una representación alternativa a la codificación estándar de potencias de 2. Bajo esta codificación los puntos adyacentes difieren en sólo un bit.

Como uno de los principales operadores genéticos, el *crossover* permite que dos individuos con alto valor de fitness o adaptabilidad puedan combinar las mejores características de cada uno.

Los nuevos individuos se crean por *crossover* alineando los padres escogidos y creando dos nuevos individuos por intercambio de subcadenas determinadas en un punto aleatorio de corte (*one-point crossover*). Otra alternativa es seleccionar 2 o más puntos de corte (*two-point* y *multi-point crossover*). También es de uso corriente el *uniform crossover* que intercambia aleatoriamente bits entre ambos padres.

En el presente trabajo se analiza la relación entre las representaciones de los individuos y los métodos de *crossover* a los cuales son sometidos, partiendo de una población inicial en la que a cada individuo se le asocia aleatoriamente una probabilidad de cruzamiento. A medida que evoluciona el (AG), las probabilidades que heredan los individuos hijos se adaptan dinámicamente en función de una regla determinística. De esta manera no solo se logra una evolución en el fitness de los individuos sino también en las probabilidades de cruzamiento.

PALABRAS CLAVE: algoritmos genéticos, adaptación, representación, crossover.

[1] Integrantes del proyecto de Investigación.

1. Introducción.

Los dos pasos más importantes al aplicar cualquier algoritmo de búsqueda heurístico a un problema particular son la especificación de la representación y la función de evaluación (fitness). Estos dos ítems constituyen el puente entre el contexto del problema original y el espacio solución del problema. En general, un espacio de búsqueda ***S*** consiste de dos subconjuntos disjuntos de subespacios de soluciones factibles y no factibles ***F*** y ***U*** respectivamente (Ver Fig.1). [1] [2]

Cuando se define un algoritmo genético (AG) es necesario elegir sus componentes tales, como operadores de variación (mutación y recombinación ó crossover), representación, mecanismo de selección de padres y una población inicial. Cada una de estas componentes tienen parámetros, por ejemplo: la probabilidad de mutación o el tamaño de la población.

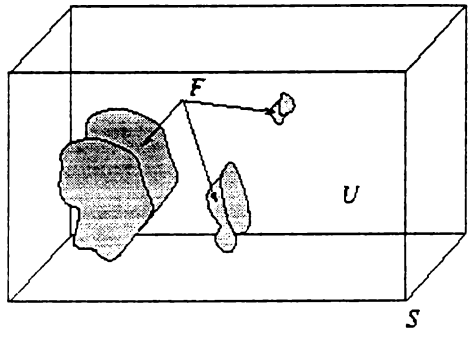

Fig. 1 Un espacio de búsqueda y sus subespacios de soluciones factibles y no factibles.

Los valores de estos parámetros determinan en gran parte si el algoritmo encontrará una solución cercana al óptimo y si la encontrará eficientemente. Elegir los valores correctos de los parámetros es una tarea que consume tiempo y esfuerzo considerable.

Los investigadores en el tema han utilizado distintas maneras para encontrar buenos valores de parámetros de forma tal que éstos afecten la performance del algoritmo significativamente. Varios investigadores han experimentado con diversos problemas de un dominio en particular, ajustando los parámetros en base a resultados de sus experimentos (ajustando 'a mano'). Luego han informado sus resultados al aplicar un (AG) en particular a un problema específico, sin una justificación de la selección realizada. En este enfoque tradicional, puede ocurrir que equivocaciones al setear los parámetros del algoritmo puedan ser origen de errores hacia sub-óptimos.

Dos importantes aportes fueron realizados por De Jong [4] y Grefenstette [5] para mejorar el diseño de un (AG). De Jong puso un considerable esfuerzo en encontrar valores de parámetros (para un (AG) tradicional) que fuesen adecuados para un conjunto de problemas de testeo numérico. Determinó experimentalmente valores recomendables para probabilidades de crossover de un punto y mutación. Sus conclusiones fueron que para sus funciones de testeo, se obtiene una performance aceptable fijando los parámetros de población en 50 individuos, probabilidad de crossover en 0.6, probabilidad de mutación en 0.001 y estrategia de selección elitista. Para otros problemas éstos valores pueden no ser tan buenos.

Por otro lado Grefenstette [5], usó un AG como un meta-algoritmo para optimizar valores de parámetros del algoritmo citados anteriormente.

El propósito es encontrar el set de parámetros óptimo y general, es decir que pueda ser aplicado a un amplio rango de problemas de optimización.

El proceso de encontrar *'a mano'* valores apropiados de parámetros y probabilidades de operadores para un AG de manera que afecten la performance del algoritmo significativamente es una tarea que implica consumo de tiempo y esfuerzo considerable.

2. Adaptación.

La acción de determinar las variables y parámetros de un (AG) para adecuarse a un problema se ha llamado 'adaptación del algoritmo' al problema, y en un (AG) esto puede hacerse mientras el algoritmo busca una solución al problema.

Los algoritmos genéticos (AGs), implementan la idea de evolución y como la evolución en sí misma puede alcanzar su actual estado de sofisticación, es natural esperar que la adaptación se utilice no solo para encontrar soluciones a un problema sino también para ajustar el algoritmo a un problema particular. Es posible modificar los parámetros durante la ejecución del AG. Ello puede hacerse usando alguna regla (posiblemente heurística), tomando información de retroalimentación del estado actual de la búsqueda o

bien empleando algún mecanismo de auto-adaptación.

Es de notar que estas modificaciones pueden afectar a un elemento de un individuo (cromosoma), a todo el individuo o más aun a toda la población. Es claro que al cambiar los valores de parámetros mientras el algoritmo está buscando la solución a un problema, se gana en eficiencia.

La auto-adaptación, basada en la evolución de la evolución, fue desarrollada en las Estrategias Evolutivas (EEs) para adaptar los parámetros de mutación durante la ejecución. El método ha sido extendido a otras áreas de la computación evolutiva pero todavía las representaciones, control de parámetros y operadores fijos, siguen siendo la norma.

Otras áreas de investigación se refieren a los mecanismos de adaptación:

- Representación de individuos.
- Operadores. Es claro que distintos operadores juegan distintos roles en distintos momentos del proceso evolutivo. Los operadores deberían adaptarse. (ej. 'Adaptative crossover, Shaffer & Morishima [6], Spears [7]).
- Control de parámetros.

En general se distinguen dos formas de *setear valores de parámetros:*

- *Ajuste de parámetros*: Se buscan buenos valores de parámetros antes de ejecutar el algoritmo. (estático)
- *Control de parámetros*: Se ejecuta el algoritmo con valores iniciales y los mismos se van cambiando a medida que se ejecuta el algoritmo. (dinámico).

En el enfoque dinámico hay distintas maneras de realizar el control. Dichas maneras se clasifican en base a dos aspectos:

- *Cómo* trabaja el mecanismo de cambio o ajuste. El tipo de mecanismo puede ser *estático o dinámico*. A su vez el dinámico puede ser *determinístico, adaptable o auto-adaptable.*
- *Qué* componentes[2] del AG se ven afectadas por el mecanismo. Es decir a qué nivel dentro del AG ocurre la adaptación. Se distinguen cuatro niveles (ambiente, población, individuos, componentes).

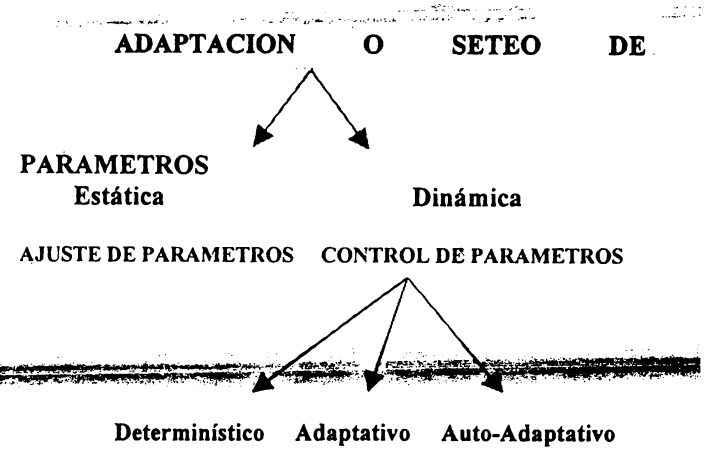

Fig. 2 Taxonomía Global de Adaptación o Seteo de Parámetros.

La clasificación del *tipo de adaptación* (Ver Fig. 2) se hace en base al mecanismo usado en el proceso, poniendo atención particular en el hecho de utilizar o no alguna información de retroalimentación del AG. (Eiben, Hinterding y Michalewicz [1] y [2]).

- Adaptación Estática.
- Adaptación Dinámica.

Para el primer caso los valores de parámetros permanecen constantes durante la ejecución del algoritmo. Se necesita de un agente externo (persona o programa) para ajustar los parámetros y elegir los valores más adecuados. Típicamente se realizan numerosas corridas de testeo tratando de encontrar la relación entre los valores de parámetros y la performance del algoritmo.

[2] 'Componentes' se refiere a partes de un AG, tales como operadores (mutación, recombinación), selección, función de fitness, etc.

La Adaptación dinámica ocurre cuando existe algún mecanismo que modifica los parámetros sin control externo. La clase de algoritmos que usan el tipo de adaptación dinámica, de acuerdo al mecanismo de adaptación pueden sub-dividirse en:

- **Adaptación dinámica determinística**: Los valores se modifican en base a una regla determinística sin usar información de retroalimentación obtenida de la ejecución del AG.
- **Adaptación dinámica adaptable**: Se usa información de retroalimentación del algortimo para determinar la dirección y/o magnitud del cambio de los parámetros. Ello incluye la acción de determinar si un valor se propaga o no en la población durante la evolución.

- **Adaptación dinámica Auto-adaptable**: En este caso se usa la idea de evolución de evolución para implementar auto-adaptación de parámetros. Los parámetros a ser adaptados se codifican como parte del individuo (cromosoma) y se lo somete a los operadores genéticos recombinación y mutación.

Experimentos.

Se implementa una versión modificada del goritmo genético canónico (Goldberg, 1990) para ıa serie de 10 experimentos (20 corridas para cada ıo) (Ver Tabla 1) en las funciones de testeo *F1: 'ichalewicz y F2:Easom* (ver Tabla 2 y Figs. 4 y .

Los sets de parámetros iniciales seleccionados .ra ambas funciones se indican en Tabla 2.

E1	Representación binaria y *crossover de un punto*
E2	Representación binaria y *crossover de dos puntos*
E3	Representación binaria y *crossover uniforme*
E4	Representación binaria y *crossover multipunto*
E5	Representación binaria y *crossover combinado (*)*
E6	Representación Gray Code *crossover de un punto*
E7	Representación Gray Code *crossover de dos puntos*
E8	Representación Gray Code *crossover uniforme*
E9	Representación Gray Code *crossover multipunto*
E10	Representación Gray Code *crossover combinado (*)*

Tabla 1. Detalle de Experimentos.

Los experimentos consistieron en aplicar resentación de individuos (cromosomas) en código binario y Gray para diferentes métodos de cruzamiento. [7], [8], [9], [10].

(*) *Crossover Combinado* consiste en aplicar aleatoriamente diferentes métodos de cruzamiento a los individuos en el pool de apareamiento.

El método utilizado para seleccionar individuos de la población para aplicarles los operadores genéticos el de *'Selección Proporcional al Fitness' o Roulette Wheel.*

Durante el proceso de evolución se realiza una adaptación o seteo de parámetros dinámica determinística. Se van modificando las probabilidades de crossover de cada individuo de la población a medida que se ejecuta el algoritmo, aplicando una regla determinística (Ver Fig.3). Se inicia el AG con una población inicial de tamaño fijo. Para cada individuo de la población, en función del experimento a realizar (E1,E2,..E10) se generan aleatoriamente las probabilidades para los distintos tipos de crossover, one point (Pc1), two point (Pc2), uniforme (Pcunif) y kpoint (Pck). Cabe aclarar que, si bien la información de probabilidades de crossover se codifican en cada individuo, no se aplica mecanismo de auto-adaptación de parámetros debido a que las probabilidades no son sometidas a operadores genéticos.

Para cada pareja de padres seleccionados del Pool De apareamiento.

1. *Elegir aleatoriamente de los padres la Probabilidad de crossover (Pc) a aplicar*

2. *Generar los dos hijos por crossover de los padres.*
3. *Evaluar los fitness de los hijos y compararlos con sus padres:*
   Si alguno de los hijos ó ambos tienen mejor fitness que sus padres asignar a los hijos la Pc elegida aleatoriamente de los padres en 1.
   Sino
   asignar al primer hijo la Pc del primer
   padre
   asignar al segundo hijo la Pc del segundo
   padre.

4. *Mutar los hijos.*

Fig. 3. Regla de selección de Probabilidad de crossover para asignar a individuos hijos.

Ej. De Aplicación de la regla (Fig. 3) para:

El Representación binaria y *crossover de un punto*

Pareja de padres seleccionados.

Cromosoma	Pc1	Pc2	Pcunif	Pck
1 0 1 1 1 0 1	0.23	0.12	0.15	0.25

Cromosoma	Pc1	Pc2	Pcunif	Pck
1 0 0 0 1 1 1	0.15	0.25	0.69	0.10

Punto de corte

Pc1 = 0.15 elegida aleatoriamente entre ambos padres.

Hijos producidos por crossover one-point en el caso de que uno o ambos hijos tengan mejor fitness que sus padres los hijos heredan la Pc1= 0.15.

Cromosoma	Pc1	Pc2	Pcunif	Pck
1 0 1 1 1 1 1	0.15	0.12	0.15	0.25

Cromosoma	Pc1	Pc2	Pcunif	Pck
1 0 0 0 1 0 1	0.15	0.25	0.69	0.10

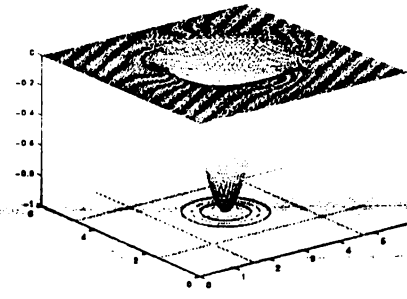

Fig. 4 Easom's Function

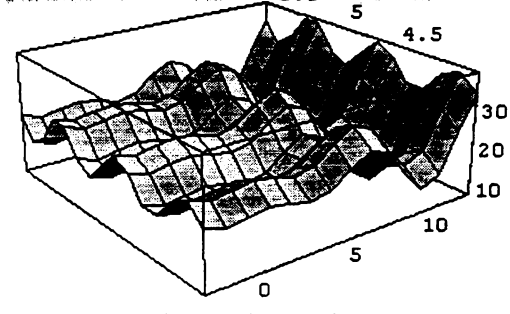

Fig. 5 Michalewicz's Function

Notación	Descripción	Set de Parámetros
F1: Michalewicz's Function (altamente multimodal)	$F1(x_1, x_2) = 21.5 + x_1 \sin(4\pi x_1) + x_2 \sin(20\pi x_2)$ Para $-3.0 <= x_1 <= 12.1$, $4.1 <= x_2 <= 5.8$ (Ver Fig. 1) Máximo valor Estimado 38.827553 (Michalewicz, 1996).	Población : 200 individuos Generaciones : 500 Probabilidad de Mutación : 0.01. Probabilidad de Crossover con adaptación.
F2:Easom's Function Unimodal, con un mínimo global en un área muy pequeña en el espacio de búsqueda.	$F2(x_1,x_2) = -\cos(x_1)\cos(x_2) e^{(-((x1-\pi)2+(x2-\pi)2))}$ Para x_1, x_2 in [-100,100] (Ver Fig. 2 a, Fig. 2 b) Valor mínimo global : -1	Población : 250 individuos Generaciones : 1000 Probabilidad de Crossover con adaptación dinámica. Probabilidad de Mutación : 0.05

Tabla 2. Funciones de Testeo F1y F2 .

Las siguientes variables de performance son izadas:

Ebest : el error porcentual del mejor individuo encontrado comparado con el máximo valor que es el valor óptimo conocido o estimado. Nos da una dea de cuan lejos estamos del óptimo. Se calcula omo:

(valor óptimo - mejor individuo)/ valor óptimo) * 100

Epop Nos da una idea de cuan lejos está el fitness romedio de la población respecto del valor ptimo estimado o conocido. Se calcula como:

(valor óptimo - fitness medio poblacional)/ valor óptimo) * 100

onclusiones.

unción de Michalewicz

En general, para los experimentos con ación dinámica de probabilidades de crossover se ieron mejores resultados que para los mentos con asignación fija de probabilidades especto a la desviación con el mejor valor trado (Ebest), y significativamente mejor para la ción con respecto a la media poblacional).

En particular el mejor comportamiento ado para las dos variables de perfomace idas (Ebest y Epop), correspondió a la estrategia ptación dinámica con representación Gray Code over combinado.

unción de Easom

En general los experimentos con adaptación ca de probabilidades de crossover tuvieron un rtamiento tan optimo como los experimentos gnación fija de probabilidades con respecto a la ión con el mejor valor encontrado (Ebest), y ativamente mejor para la desviación con o a la media poblacional (Epop).

En particular el mejor comportamiento ido para las dos variables de perfomace las (Ebest y Epop), correspondió a la estrategia tación dinámica con representación Binaria y er combinado.

encias.

Hinterding, R., Michalewicz, Z., and Eiben, A.E., *Adaptation in Evolutionary Computation: A Survey,* Proceedings of the 4th IEEE International Conference on Evolutionary Computation, Indianapolis, April 13-16, 1997. pp.65-69.

[2] Eiben, A.E., Hinterding, R., and Michalewicz, Z., *Parameter Control in Evolutionary Algorithms,* IEEE Transactions on Evolutionary Computation, Vol.3, No.2, 1999.

[3] Spears, William M. (1992). *Adapting Crossover in a Genetic Algorithm* (Technical Report AIC-92-025). Washington, DC: Naval Research aboratory,Navy Center for Applied Research on Artificial Intelligence.

[4] K. De Jong. *'The Analysis of the Behavior of a class of Genetic Adaptive Systems'* PhD. thesis, Department of Computer Science, University of Michigan, Ann Arbor, Michigan, 1975.

[5] J.J. Grefenstette. *'Optimization of control parameters for genetic algorithms.* IEEE Transactions on Systems, Many and Cybernetics, 16 (1):122-128 1988.

[6] J.D. Schaffer and A. Morishima. *'An adaptive crossover distribution mechanism for genetic algorithms'. In Proceedings of the 2nd International Conference on Genetic Algorithms.* Laurence Erlbaum Associates, 1987. Pp, 38-40.

[7] W.M. Spears. 'Adapting Crossover in evolutionary algorithms. In J.R. Mc. Donnell, R.G. Reynolds, and D.B. Fogel, editors, *Proceedings of the Fourth Annual Conference on Evolutionary Programming,* The MIT Press, 1995, pp. 367-384

[8] Golberg, D.E, *"Genetic Algorithms in Search Optimization & Machine Learning'* Addison-Wesley Publishing Company. (1.988).

[9] Michalewicz, M: *"Genetic Algorithms + Data Structures = Evolution Programs"*. Springer third revised edition, 1.996

[10] Mathias K, Whitley L.D, *"Changing representations during Search: A Comparative Study of Delta Coding"* - Evolutionary Computation 2 (3): 249-278. © The Massachusetts Institute of Technology.